615 Assignment Strawberries 3

Haoran Cui

2024-10-02

#Preparing data for analysis —— Strawberries

```r
library(knitr)
library(kableExtra)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()     masks stats::filter()
## x dplyr::group_rows() masks kableExtra::group_rows()
## x dplyr::lag()        masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(dplyr)
library(readr)
library(tidyr)
library(stringr)
library(ggplot2)
```

```r
# Load the data from a CSV file and view the first few rows
strawberry <- read_csv("strawberries25_v3.csv", col_names = TRUE)
```

```
## Rows: 12669 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (15): Program, Period, Geo Level, State, State ANSI, Ag District, County...
## dbl  (2): Year, Ag District Code
## lgl  (4): Week Ending, Zip Code, Region, Watershed
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(strawberry)
```

```
## # A tibble: 6 x 21
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## # i 14 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>
```

```
# Replace any occurrences of "(D)" in Value and CV% columns with NA (missing value)
strawberry <- strawberry %>%
  mutate(
    Value = ifelse(Value == "(D)", NA, Value),
    `CV (%)` = ifelse(`CV (%)` == "(D)", NA, `CV (%)`)
  )
head(strawberry)
```

```
## # A tibble: 6 x 21
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## # i 14 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>
```

```
# Rearrange 'Domain' column into three new columns: chemical category, name, and number
strawberry <- strawberry %>%
  mutate(
    Category = case_when(
      Domain == "Total" ~ NA_character_,  # If Domain is "Total", mark as NA
      str_detect(Domain, "CHEMICAL") ~ str_trim(str_remove(Domain, "CHEMICAL, ")),  # Remove "CHEMICAL,
      TRUE ~ Domain
    )
  )
unique(strawberry$Category)
```

```
## [1] "TOTAL"           "AREA GROWN"      "ORGANIC STATUS" "FUNGICIDE"
## [5] "INSECTICIDE"     "OTHER"           "HERBICIDE"      "FERTILIZER"
```

```
head(strawberry)
```

```
## # A tibble: 6 x 22
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## # i 15 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>,
## #   Category <chr>
```

```
# Extract "Name" and "Number" from the 'Domain Category' column
strawberry <- strawberry %>%
  mutate(
    Name = case_when(
      Category == "TOTAL" ~ NA_character_,  # If Category is "TOTAL", mark as NA
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*=.*\\)") ~
        str_extract(`Domain Category`, "(?<=\\().*?(?=\\s?=)"),  # Extract Name from Domain Category
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*\\)") ~
        str_extract(`Domain Category`, "(?<=\\().*?(?=\\))"),  # Another pattern for extraction
      TRUE ~ NA_character_
    ),
    Number = case_when(
      Category == "TOTAL" ~ NA_real_,  # If Category is "TOTAL", mark as NA
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*=.*\\)") ~
        as.numeric(str_extract(`Domain Category`, "(?<=\\=\\s?).*?(?=\\))")),  # Extract Number from Do
      str_detect(`Domain Category`, fixed(Category)) & str_detect(`Domain Category`, "\\(.*\\)") ~
        NA_real_,  # If no number, mark as NA
      TRUE ~ NA_real_
    )
  )
```

```
strawberry <- strawberry %>%
  mutate(Category = case_when(
    `Domain Category` == "NOT SPECIFIED" ~ NA_character_,  # If Domain Category is "NOT SPECIFIED", mar
    TRUE ~ Category  # Otherwise, retain the existing Category
  ))
head(strawberry)
```

```
## # A tibble: 6 x 24
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## # i 17 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>,
## #   Category <chr>, Name <chr>, Number <dbl>
```

```
# Clean and extract numerical intervals for planted area, creating Min and Max columns
strawberry <- strawberry %>%
  mutate(
    Min = case_when(
      str_detect(Name, "100 OR MORE ACRES") ~ 100,  # If the text says "100 OR MORE ACRES", Min is 100
      str_detect(Name, "TO") ~ as.numeric(str_extract(Name, "^[0-9.]+")),  # Extract Min value from int
      TRUE ~ NA_real_
    ),
    Max = case_when(
      str_detect(Name, "100 OR MORE ACRES") ~ "MORE",  # For "100 OR MORE ACRES", Max is "MORE"
      str_detect(Name, "TO") ~ str_extract(Name, "(?<=TO )^[0-9.]+"),  # Extract Max value from interva
      TRUE ~ NA_character_
    )
  )

# View the cleaned data
head(strawberry)
```

```
## # A tibble: 6 x 26
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## # i 19 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>,
## #   Category <chr>, Name <chr>, Number <dbl>, Min <dbl>, Max <chr>
```

```
# Extract 'Unit' from the 'Data Item' column (substring after "MEASURED")
strawberry <- strawberry %>%
  mutate(Unit = str_extract(strawberry$`Data Item`, "(?<=MEASURED ).*"))

# Extract 'Type' by identifying either "BEARING" or "ORGANIC" in the 'Data Item' column
strawberry <- strawberry %>%
  mutate(Type = str_extract(strawberry$`Data Item`, "BEARING|ORGANIC"))

# Extract 'Operation' by removing 'MEASURED', 'BEARING', and 'ORGANIC'
strawberry <- strawberry %>%
  mutate(Operation = str_replace_all(strawberry$`Data Item`, "MEASURED.*|BEARING|ORGANIC", "") %>%
           str_trim())

# Further clean 'Operation' by removing additional terms ('STRAWBERRIES', commas, hyphens)
strawberry <- strawberry %>%
  mutate(Operation = str_replace_all(strawberry$`Data Item`, "MEASURED.*|BEARING|ORGANIC|STRAWBERRIES(,
           str_replace_all("[-,]", "") %>%
           str_trim())

# View the resulting data
head(strawberry)
```

```
## # A tibble: 6 x 29
##   Program  Year Period `Week Ending` `Geo Level` State   `State ANSI`
##   <chr>   <dbl> <chr>  <lgl>         <chr>       <chr>   <chr>
## 1 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 2 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 3 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 4 CENSUS   2022 YEAR   NA            COUNTY      ALABAMA 01
## 5 CENSUS   2022 YEAR   NA             COUNTY      ALABAMA 01
## 6 CENSUS   2022 YEAR   NA             COUNTY      ALABAMA 01
## # i 22 more variables: `Ag District` <chr>, `Ag District Code` <dbl>,
## #   County <chr>, `County ANSI` <chr>, `Zip Code` <lgl>, Region <lgl>,
## #   watershed_code <chr>, Watershed <lgl>, Commodity <chr>, `Data Item` <chr>,
## #   Domain <chr>, `Domain Category` <chr>, Value <chr>, `CV (%)` <chr>,
## #   Category <chr>, Name <chr>, Number <dbl>, Min <dbl>, Max <chr>, Unit <chr>,
## #   Type <chr>, Operation <chr>
```

```r
# Export the cleaned data to a CSV file
write.csv(strawberry, "cleaned_strawberries.csv", row.names = FALSE)
```

```r
# Check the structure of the cleaned dataset
str(strawberry)
```

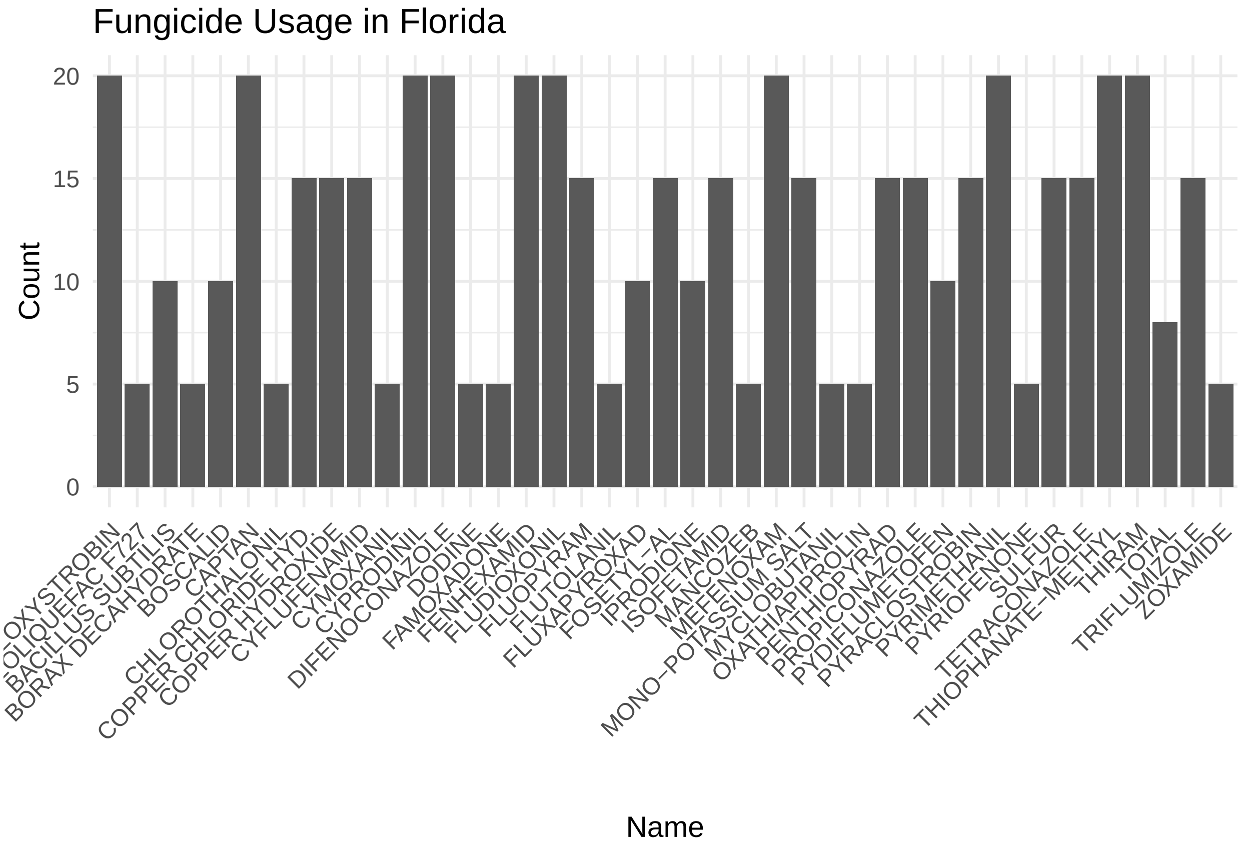
```
## tibble [12,669 x 29] (S3: tbl_df/tbl/data.frame)
##  $ Program         : chr [1:12669] "CENSUS" "CENSUS" "CENSUS" "CENSUS" ...
##  $ Year            : num [1:12669] 2022 2022 2022 2022 2022 ...
##  $ Period          : chr [1:12669] "YEAR" "YEAR" "YEAR" "YEAR" ...
##  $ Week Ending     : logi [1:12669] NA NA NA NA NA NA ...
##  $ Geo Level       : chr [1:12669] "COUNTY" "COUNTY" "COUNTY" "COUNTY" ...
##  $ State           : chr [1:12669] "ALABAMA" "ALABAMA" "ALABAMA" "ALABAMA" ...
##  $ State ANSI      : chr [1:12669] "01" "01" "01" "01" ...
##  $ Ag District     : chr [1:12669] "BLACK BELT" "BLACK BELT" "BLACK BELT" "BLACK BELT" ...
##  $ Ag District Code: num [1:12669] 40 40 40 40 40 40 40 40 40 40 ...
##  $ County          : chr [1:12669] "BULLOCK" "BULLOCK" "BULLOCK" "BULLOCK" ...
##  $ County ANSI     : chr [1:12669] "011" "011" "011" "011" ...
##  $ Zip Code        : logi [1:12669] NA NA NA NA NA NA ...
##  $ Region          : logi [1:12669] NA NA NA NA NA NA ...
##  $ watershed_code  : chr [1:12669] "00000000" "00000000" "00000000" "00000000" ...
##  $ Watershed       : logi [1:12669] NA NA NA NA NA NA ...
##  $ Commodity       : chr [1:12669] "STRAWBERRIES" "STRAWBERRIES" "STRAWBERRIES" "STRAWBERRIES" ...
##  $ Data Item       : chr [1:12669] "STRAWBERRIES - ACRES BEARING" "STRAWBERRIES - ACRES GROWN" "STRA
##  $ Domain          : chr [1:12669] "TOTAL" "TOTAL" "TOTAL" "TOTAL" ...
##  $ Domain Category : chr [1:12669] "NOT SPECIFIED" "NOT SPECIFIED" "NOT SPECIFIED" "NOT SPECIFIED" .
##  $ Value           : chr [1:12669] NA "3" NA "1" ...
##  $ CV (%)          : chr [1:12669] NA "15.7" NA "(L)" ...
##  $ Category        : chr [1:12669] NA NA NA NA ...
##  $ Name            : chr [1:12669] NA NA NA NA ...
##  $ Number          : num [1:12669] NA NA NA NA NA NA NA NA NA NA ...
##  $ Min             : num [1:12669] NA NA NA NA NA NA NA NA NA NA ...
##  $ Max             : chr [1:12669] NA NA NA NA ...
##  $ Unit            : chr [1:12669] NA NA NA NA ...
##  $ Type            : chr [1:12669] "BEARING" NA "BEARING" "BEARING" ...
##  $ Operation       : chr [1:12669] "ACRES" "ACRES GROWN" "ACRES NON" "OPERATIONS WITH AREA" ...
```

cleaned_strawberries.csv

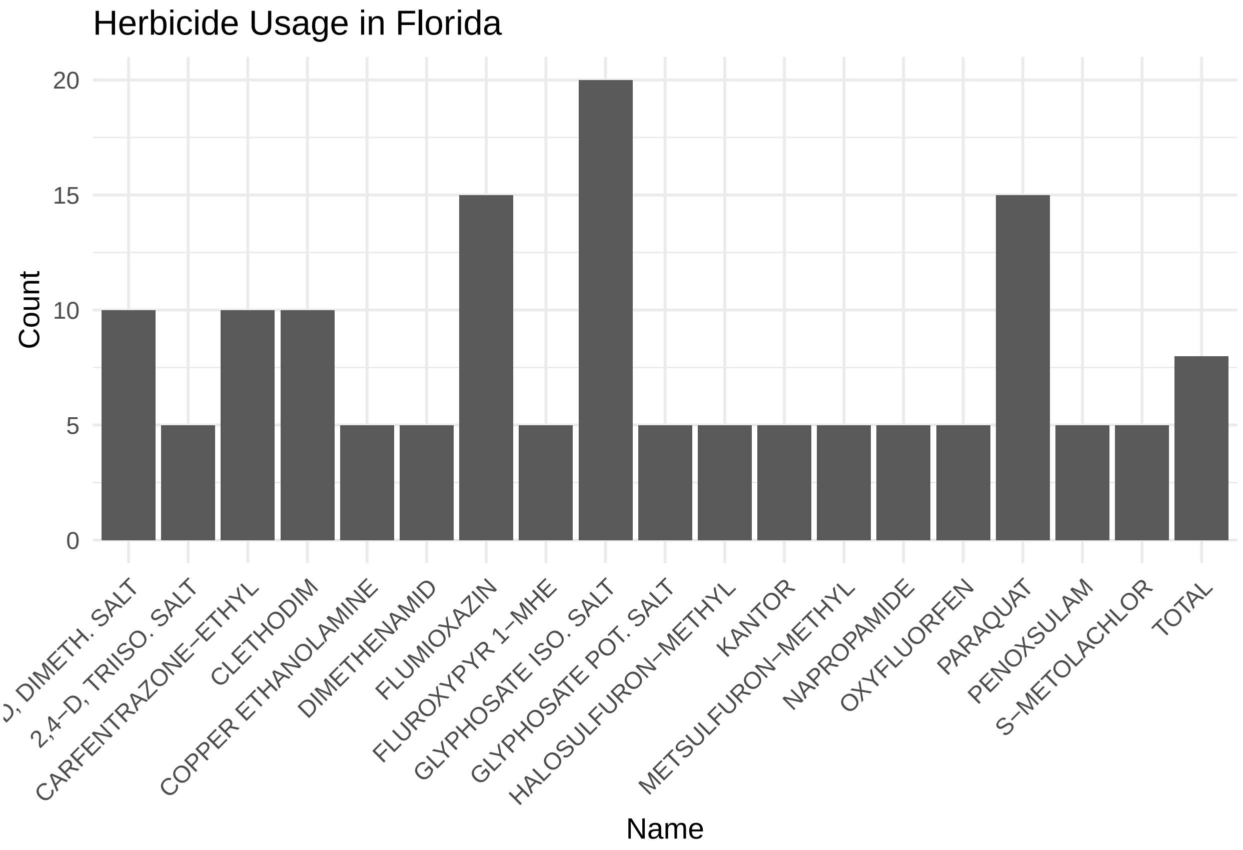
```r
strawberries <- read.csv("cleaned_strawberries.csv")

# Function to filter by category, state, and group by Name
filter_and_group <- function(data, category) {
  filtered_data <- subset(data, Category == category & State == "FLORIDA")
  grouped_data <- split(filtered_data, filtered_data$Name) # Group by Name
  return(grouped_data)
}
```

```r
# Apply the function to each category
fungicide_florida_grouped <- filter_and_group(strawberries, "FUNGICIDE")
herbicide_florida_grouped <- filter_and_group(strawberries, "HERBICIDE")
insecticide_florida_grouped <- filter_and_group(strawberries, "INSECTICIDE")
other_florida_grouped <- filter_and_group(strawberries, "OTHER")
```

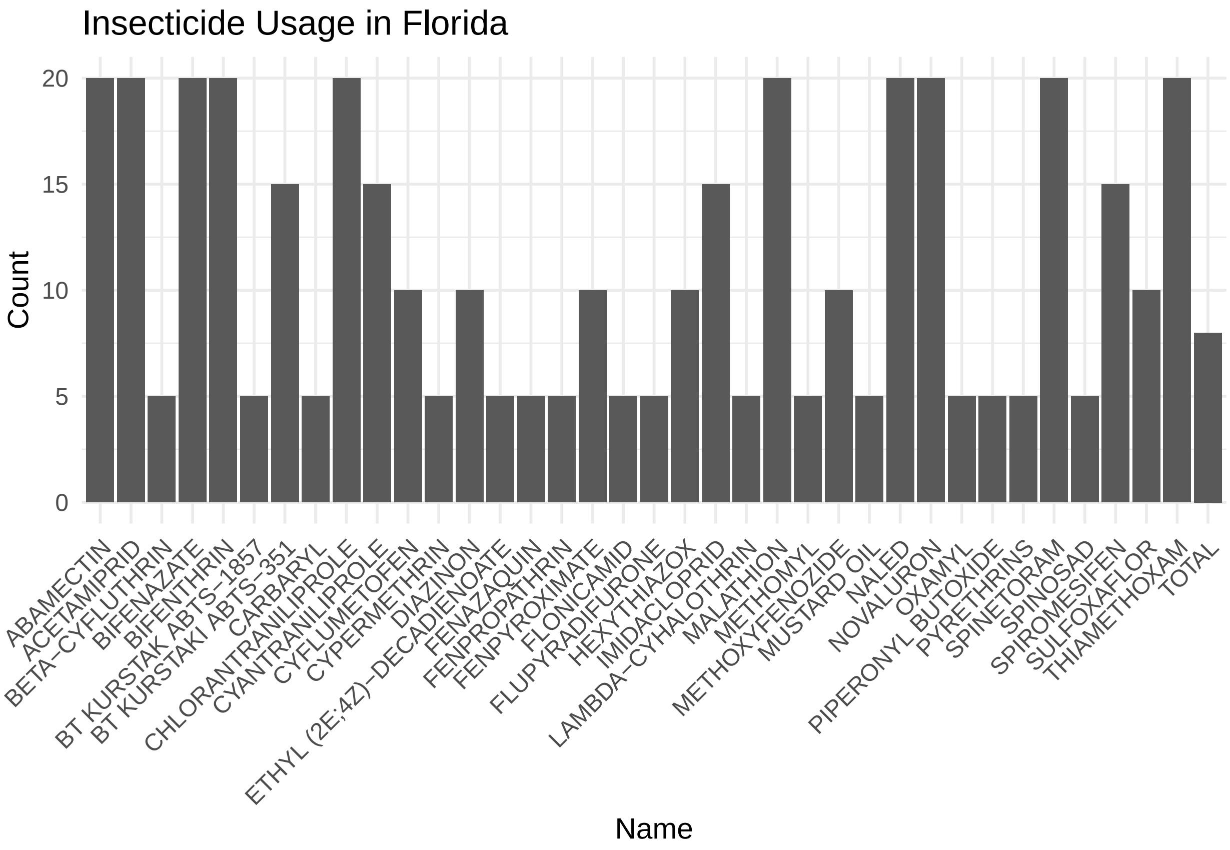
```r
library(ggplot2)

# Function to create bar plots for each group
visualize_grouped_data <- function(grouped_data, title) {
  # Combine the data for easier plotting
  combined_data <- do.call(rbind, grouped_data)

  # Create a bar plot
  ggplot(combined_data, aes(x = Name)) +
    geom_bar() +
    labs(title = title, x = "Name", y = "Count") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))
}

# Visualize each category
visualize_grouped_data(fungicide_florida_grouped, "Fungicide Usage in Florida")
```



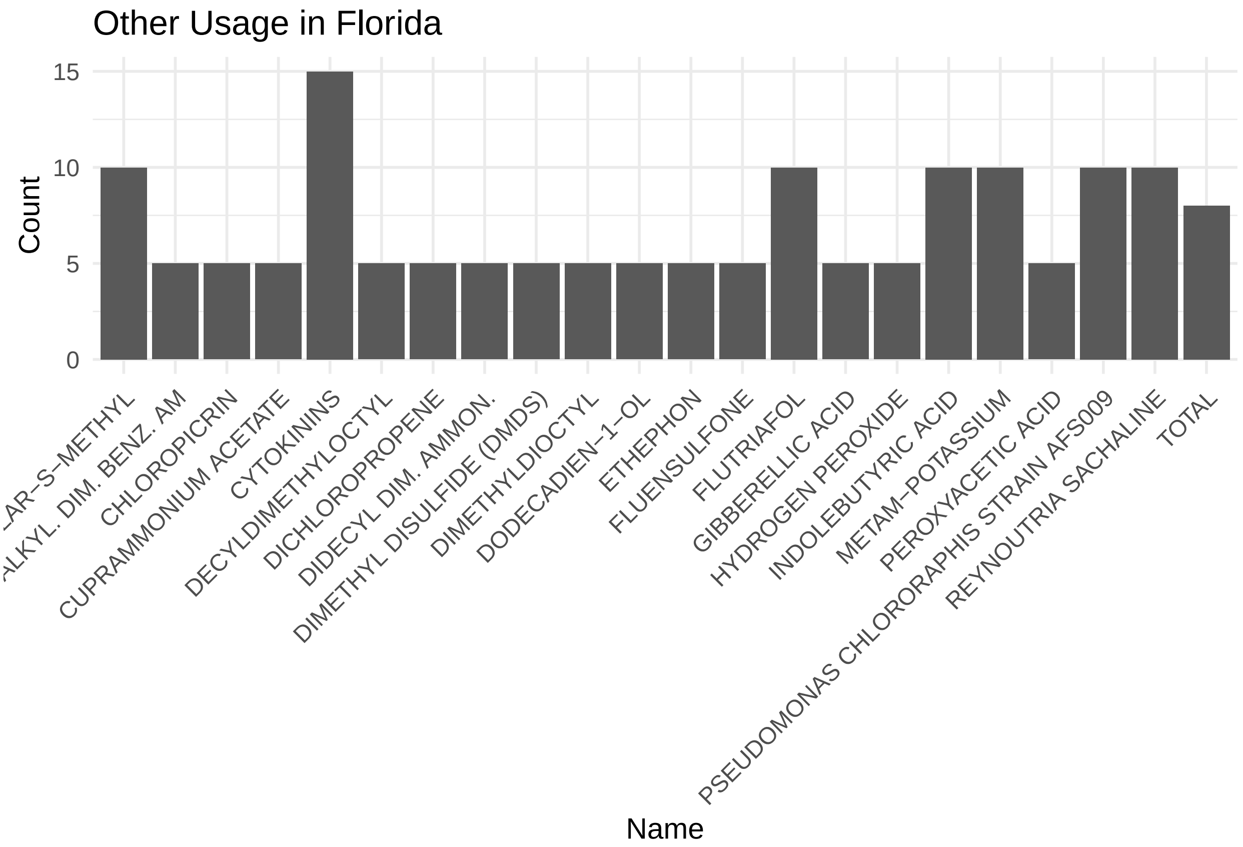
```r
visualize_grouped_data(herbicide_florida_grouped, "Herbicide Usage in Florida")
```



```r
visualize_grouped_data(insecticide_florida_grouped, "Insecticide Usage in Florida")
```



```r
visualize_grouped_data(other_florida_grouped, "Other Usage in Florida")
```



Plot Explanation: 1. Each plot displays the count of various chemicals used in a specific category (fungicide, herbicide, etc.). 2. The x-axis lists different chemical names. 3. The y-axis represents the count of each chemical. 4. The bars represent the usage count of each chemical in that category.

Title and Axes: 1. Each plot has a title specifying the type of chemical usage (e.g., "Fungicide Usage in Florida"). 2. Chemical names are displayed on the x-axis, rotated at a 45-degree angle to fit them within the plot. 3. The y-axis shows the count of each chemical.

Each of the four plots represents the chemical usage data for a specific category, helping to compare the usage of different chemicals in each category

```r
# Function to filter by category and state, then find the most and least frequent Name
find_most_least_frequent <- function(data, category) {
  # Filter data by category and state (Florida)
  filtered_data <- subset(data, Category == category & State == "FLORIDA")

  # Count occurrences of each Name
  name_counts <- table(filtered_data$Name)

  # Find the most frequent Name
  most_frequent <- names(name_counts[name_counts == max(name_counts)])

  # Find the least frequent Name
  least_frequent <- names(name_counts[name_counts == min(name_counts)])

  return(list("most_frequent" = most_frequent, "least_frequent" = least_frequent))
}

# Apply the function to each category
```

```
fungicide_florida_freq <- find_most_least_frequent(strawberries, "FUNGICIDE")
herbicide_florida_freq <- find_most_least_frequent(strawberries, "HERBICIDE")
insecticide_florida_freq <- find_most_least_frequent(strawberries, "INSECTICIDE")
other_florida_freq <- find_most_least_frequent(strawberries, "OTHER")

# Print the results for each category
print("Fungicide:")
```

```
## [1] "Fungicide:"
```

```
print(fungicide_florida_freq)
```

```
## $most_frequent
##  [1] "AZOXYSTROBIN"       "CAPTAN"             "CYPRODINIL"        
##  [4] "DIFENOCONAZOLE"     "FENHEXAMID"         "FLUDIOXONIL"       
##  [7] "MEFENOXAM"          "PYRIMETHANIL"       "THIOPHANATE-METHYL"
## [10] "THIRAM"            
## 
## $least_frequent
##  [1] "BACILLUS AMYLOLIQUEFAC F727" "BORAX DECAHYDRATE"          
##  [3] "CHLOROTHALONIL"              "CYMOXANIL"                  
##  [5] "DODINE"                      "FAMOXADONE"                 
##  [7] "FLUTOLANIL"                  "MANCOZEB"                   
##  [9] "MYCLOBUTANIL"                "OXATHIAPIPROLIN"            
## [11] "PYRIOFENONE"                 "ZOXAMIDE"
```

```
print("Herbicide:")
```

```
## [1] "Herbicide:"
```

```
print(herbicide_florida_freq)
```

```
## $most_frequent
## [1] "GLYPHOSATE ISO. SALT"
## 
## $least_frequent
##  [1] "2,4-D, TRIISO. SALT"  "COPPER ETHANOLAMINE"  "DIMETHENAMID"        
##  [4] "FLUROXYPYR 1-MHE"     "GLYPHOSATE POT. SALT" "HALOSULFURON-METHYL" 
##  [7] "KANTOR"               "METSULFURON-METHYL"   "NAPROPAMIDE"         
## [10] "OXYFLUORFEN"          "PENOXSULAM"           "S-METOLACHLOR"
```

```
print("Insecticide:")
```

```
## [1] "Insecticide:"
```

```
print(insecticide_florida_freq)
```

```
## $most_frequent
##  [1] "ABAMECTIN"           "ACETAMIPRID"         "BIFENAZATE"         
##  [4] "BIFENTHRIN"          "CHLORANTRANILIPROLE" "MALATHION"          
##  [7] "NALED"               "NOVALURON"           "SPINETORAM"         
## [10] "THIAMETHOXAM"       
## 
## $least_frequent
##  [1] "BETA-CYFLUTHRIN"            "BT KURSTAK ABTS-1857"      
##  [3] "CARBARYL"                   "CYPERMETHRIN"              
##  [5] "ETHYL (2E;4Z)-DECADIENOATE" "FENAZAQUIN"
##  [7] "FENPROPATHRIN"              "FLONICAMID"
##  [9] "FLUPYRADIFURONE"            "LAMBDA-CYHALOTHRIN"
## [11] "METHOMYL"                   "MUSTARD OIL"
## [13] "OXAMYL"                     "PIPERONYL BUTOXIDE"
## [15] "PYRETHRINS"                 "SPINOSAD"
```

```r
print("Other:")
```

```
## [1] "Other:"
```

```r
print(other_florida_freq)
```

```
## $most_frequent
## [1] "CYTOKININS"
##
## $least_frequent
##  [1] "ALKYL. DIM. BENZ. AM"      "CHLOROPICRIN"
##  [3] "CUPRAMMONIUM ACETATE"      "DECYLDIMETHYLOCTYL"
##  [5] "DICHLOROPROPENE"           "DIDECYL DIM. AMMON."
##  [7] "DIMETHYL DISULFIDE (DMDS)" "DIMETHYLDIOCTYL"
##  [9] "DODECADIEN-1-OL"           "ETHEPHON"
## [11] "FLUENSULFONE"              "GIBBERELLIC ACID"
## [13] "HYDROGEN PEROXIDE"         "PEROXYACETIC ACID"
```

```r
library(tidyverse)
library(PubChemR)

# Function to retrieve the GHS hazard statements with error handling
GHS_searcher <- function(result_json_object) {
  # Check if 'result', 'Hierarchies', and 'Hierarchy' exist and are not null
  if (!is.null(result_json_object[["result"]]) &&
      !is.null(result_json_object[["result"]][["Hierarchies"]]) &&
      !is.null(result_json_object[["result"]][["Hierarchies"]][["Hierarchy"]])) {

    hierarchy_list <- result_json_object[["result"]][["Hierarchies"]][["Hierarchy"]]

    # Loop through the hierarchy list and check for the GHS Classification
    for (i in seq_along(hierarchy_list)) {
      if (!is.null(hierarchy_list[[i]][["SourceName"]]) &&
          hierarchy_list[[i]][["SourceName"]] == "GHS Classification (UNECE)") {
        return(i)  # Return the index where GHS Classification is found
      }
    }
  }

  # If no GHS classification is found, return NA
  return(NA)
}

# Function to retrieve hazard details from the hierarchy with error handling
hazards_retriever <- function(index, result_json_object) {
  if (!is.na(index)) {
    hierarchy <- result_json_object[["result"]][["Hierarchies"]][["Hierarchy"]][[index]]
    if (!is.null(hierarchy[["Node"]])) {
      i <- 1
      output_list <- rep(NA, length(hierarchy[["Node"]]))
      while (i <= length(hierarchy[["Node"]]) &&
             !is.null(hierarchy[["Node"]][[i]][["Information"]][["Name"]]) &&
             str_detect(hierarchy[["Node"]][[i]][["Information"]][["Name"]], "H")) {
        output_list[i] <- hierarchy[["Node"]][[i]][["Information"]][["Name"]]
        i <- i + 1
      }

      return(output_list[!is.na(output_list)])  # Return non-NA hazard statements
    }
  }

  return(paste("No hazard information found"))
}

# Function to fetch and print hazard statements for a chemical
fetch_hazard_statements <- function(chemical_name) {
  result <- get_pug_rest(identifier = chemical_name, namespace = "name", domain = "compound", operation
  index <- GHS_searcher(result)
  if (!is.na(index)) {
    hazards <- hazards_retriever(index, result)
    return(hazards)
  } else {
    return(paste("No GHS classification found for", chemical_name))
  }
}

# Function to filter by category and state, then find the most and least frequent Name
find_most_least_frequent <- function(data, category) {
  # Filter data by category and state (Florida)
  filtered_data <- subset(data, Category == category & State == "FLORIDA")

  # Count occurrences of each Name
  name_counts <- table(filtered_data$Name)

  # Find the most frequent Name
  most_frequent <- names(name_counts[name_counts == max(name_counts)])

  # Find the least frequent Name
  least_frequent <- names(name_counts[name_counts == min(name_counts)])

  return(list("most_frequent" = most_frequent, "least_frequent" = least_frequent))
}

# Assuming 'strawberries' data has already been loaded
# Retrieve the most and least frequent chemicals for each group
fungicide_florida_freq <- find_most_least_frequent(strawberries, "FUNGICIDE")
herbicide_florida_freq <- find_most_least_frequent(strawberries, "HERBICIDE")
insecticide_florida_freq <- find_most_least_frequent(strawberries, "INSECTICIDE")
other_florida_freq <- find_most_least_frequent(strawberries, "OTHER")

categories <- list(
  "Fungicide" = fungicide_florida_freq,
  "Herbicide" = herbicide_florida_freq,
```

```r
  "Insecticide" = insecticide_florida_freq,
  "Other" = other_florida_freq
)

# Loop through each category to get hazard statements for the most and least frequent chemicals
for (category in names(categories)) {
  cat(paste("\nCategory:", category, "\n"))

  # Most frequent chemical
  most_frequent <- categories[[category]]$most_frequent
  cat(paste("Most frequent chemical:", most_frequent, "\n"))
  most_hazards <- fetch_hazard_statements(most_frequent)
  print(most_hazards)

  # Least frequent chemical
  least_frequent <- categories[[category]]$least_frequent
  cat(paste("Least frequent chemical:", least_frequent, "\n"))
  least_hazards <- fetch_hazard_statements(least_frequent)
  print(least_hazards)
}
```

```
##
## Category: Fungicide
## Most frequent chemical: AZOXYSTROBIN
##  Most frequent chemical: CAPTAN
##  Most frequent chemical: CYPRODINIL
##  Most frequent chemical: DIFENOCONAZOLE
##  Most frequent chemical: FENHEXAMID
##  Most frequent chemical: FLUDIOXONIL
##  Most frequent chemical: MEFENOXAM
##  Most frequent chemical: PYRIMETHANIL
##  Most frequent chemical: THIOPHANATE-METHYL
##  Most frequent chemical: THIRAM
```

```
## Request failed [404]. Retrying in 3.8 seconds...
```

```
## Request failed [404]. Retrying in 1 seconds...
```

```
##  [1] "No GHS classification found for AZOXYSTROBIN"
##  [2] "No GHS classification found for CAPTAN"
##  [3] "No GHS classification found for CYPRODINIL"
##  [4] "No GHS classification found for DIFENOCONAZOLE"
##  [5] "No GHS classification found for FENHEXAMID"
##  [6] "No GHS classification found for FLUDIOXONIL"
##  [7] "No GHS classification found for MEFENOXAM"
##  [8] "No GHS classification found for PYRIMETHANIL"
##  [9] "No GHS classification found for THIOPHANATE-METHYL"
## [10] "No GHS classification found for THIRAM"
## Least frequent chemical: BACILLUS AMYLOLIQUEFAC F727
##  Least frequent chemical: BORAX DECAHYDRATE
##  Least frequent chemical: CHLOROTHALONIL
##  Least frequent chemical: CYMOXANIL
##  Least frequent chemical: DODINE
##  Least frequent chemical: FAMOXADONE
##  Least frequent chemical: FLUTOLANIL
##  Least frequent chemical: MANCOZEB
##  Least frequent chemical: MYCLOBUTANIL
##  Least frequent chemical: OXATHIAPIPROLIN
##  Least frequent chemical: PYRIOFENONE
##  Least frequent chemical: ZOXAMIDE

## Request failed [404]. Retrying in 1 seconds...

## Request failed [404]. Retrying in 5.5 seconds...

##  [1] "No GHS classification found for BACILLUS AMYLOLIQUEFAC F727"
##  [2] "No GHS classification found for BORAX DECAHYDRATE"
##  [3] "No GHS classification found for CHLOROTHALONIL"
##  [4] "No GHS classification found for CYMOXANIL"
##  [5] "No GHS classification found for DODINE"
##  [6] "No GHS classification found for FAMOXADONE"
##  [7] "No GHS classification found for FLUTOLANIL"
##  [8] "No GHS classification found for MANCOZEB"
##  [9] "No GHS classification found for MYCLOBUTANIL"
## [10] "No GHS classification found for OXATHIAPIPROLIN"
## [11] "No GHS classification found for PYRIOFENONE"
## [12] "No GHS classification found for ZOXAMIDE"
## 
## Category: Herbicide
## Most frequent chemical: GLYPHOSATE ISO. SALT

## Request failed [404]. Retrying in 3.4 seconds...

## Request failed [404]. Retrying in 4 seconds...

## [1] "No GHS classification found for GLYPHOSATE ISO. SALT"
## Least frequent chemical: 2,4-D, TRIISO. SALT
##  Least frequent chemical: COPPER ETHANOLAMINE
##  Least frequent chemical: DIMETHENAMID
##  Least frequent chemical: FLUROXYPYR 1-MHE
##  Least frequent chemical: GLYPHOSATE POT. SALT
##  Least frequent chemical: HALOSULFURON-METHYL
##  Least frequent chemical: KANTOR
##  Least frequent chemical: METSULFURON-METHYL
##  Least frequent chemical: NAPROPAMIDE
##  Least frequent chemical: OXYFLUORFEN
##  Least frequent chemical: PENOXSULAM
##  Least frequent chemical: S-METOLACHLOR

## Request failed [404]. Retrying in 1.6 seconds...

## Request failed [404]. Retrying in 7.6 seconds...

##  [1] "No GHS classification found for 2,4-D, TRIISO. SALT"
##  [2] "No GHS classification found for COPPER ETHANOLAMINE"
##  [3] "No GHS classification found for DIMETHENAMID"
##  [4] "No GHS classification found for FLUROXYPYR 1-MHE"
##  [5] "No GHS classification found for GLYPHOSATE POT. SALT"
##  [6] "No GHS classification found for HALOSULFURON-METHYL"
##  [7] "No GHS classification found for KANTOR"
##  [8] "No GHS classification found for METSULFURON-METHYL"
##  [9] "No GHS classification found for NAPROPAMIDE"
## [10] "No GHS classification found for OXYFLUORFEN"
## [11] "No GHS classification found for PENOXSULAM"
## [12] "No GHS classification found for S-METOLACHLOR"
##
## Category: Insecticide
## Most frequent chemical: ABAMECTIN
##  Most frequent chemical: ACETAMIPRID
##  Most frequent chemical: BIFENAZATE
##  Most frequent chemical: BIFENTHRIN
##  Most frequent chemical: CHLORANTRANILIPROLE
##  Most frequent chemical: MALATHION
##  Most frequent chemical: NALED
##  Most frequent chemical: NOVALURON
##  Most frequent chemical: SPINETORAM
##  Most frequent chemical: THIAMETHOXAM
```

```
## Request failed [404]. Retrying in 3 seconds...
```

```
## Request failed [404]. Retrying in 1.9 seconds...
```

```
##  [1] "No GHS classification found for ABAMECTIN"
##  [2] "No GHS classification found for ACETAMIPRID"
##  [3] "No GHS classification found for BIFENAZATE"
##  [4] "No GHS classification found for BIFENTHRIN"
##  [5] "No GHS classification found for CHLORANTRANILIPROLE"
##  [6] "No GHS classification found for MALATHION"
##  [7] "No GHS classification found for NALED"
##  [8] "No GHS classification found for NOVALURON"
##  [9] "No GHS classification found for SPINETORAM"
## [10] "No GHS classification found for THIAMETHOXAM"
## Least frequent chemical: BETA-CYFLUTHRIN
##  Least frequent chemical: BT KURSTAK ABTS-1857
##  Least frequent chemical: CARBARYL
##  Least frequent chemical: CYPERMETHRIN
##  Least frequent chemical: ETHYL (2E;4Z)-DECADIENOATE
##  Least frequent chemical: FENAZAQUIN
##  Least frequent chemical: FENPROPATHRIN
##  Least frequent chemical: FLONICAMID
##  Least frequent chemical: FLUPYRADIFURONE
##  Least frequent chemical: LAMBDA-CYHALOTHRIN
##  Least frequent chemical: METHOMYL
##  Least frequent chemical: MUSTARD OIL
##  Least frequent chemical: OXAMYL
##  Least frequent chemical: PIPERONYL BUTOXIDE
##  Least frequent chemical: PYRETHRINS
##  Least frequent chemical: SPINOSAD
```

```
## Request failed [404]. Retrying in 3.8 seconds...
```

```
## Request failed [404]. Retrying in 5.8 seconds...
```

```
##  [1] "No GHS classification found for BETA-CYFLUTHRIN"
##  [2] "No GHS classification found for BT KURSTAK ABTS-1857"
##  [3] "No GHS classification found for CARBARYL"
##  [4] "No GHS classification found for CYPERMETHRIN"
##  [5] "No GHS classification found for ETHYL (2E;4Z)-DECADIENOATE"
##  [6] "No GHS classification found for FENAZAQUIN"
##  [7] "No GHS classification found for FENPROPATHRIN"
##  [8] "No GHS classification found for FLONICAMID"
##  [9] "No GHS classification found for FLUPYRADIFURONE"
## [10] "No GHS classification found for LAMBDA-CYHALOTHRIN"
## [11] "No GHS classification found for METHOMYL"
## [12] "No GHS classification found for MUSTARD OIL"
## [13] "No GHS classification found for OXAMYL"
## [14] "No GHS classification found for PIPERONYL BUTOXIDE"
## [15] "No GHS classification found for PYRETHRINS"
## [16] "No GHS classification found for SPINOSAD"
##
## Category: Other
## Most frequent chemical: CYTOKININS
```

```
## Request failed [404]. Retrying in 1.3 seconds...
```

```
## Request failed [404]. Retrying in 4.9 seconds...
```

```
## [1] "No GHS classification found for CYTOKININS"
## Least frequent chemical: ALKYL. DIM. BENZ. AM
##  Least frequent chemical: CHLOROPICRIN
##  Least frequent chemical: CUPRAMMONIUM ACETATE
##  Least frequent chemical: DECYLDIMETHYLOCTYL
##  Least frequent chemical: DICHLOROPROPENE
##  Least frequent chemical: DIDECYL DIM. AMMON.
##  Least frequent chemical: DIMETHYL DISULFIDE (DMDS)
##  Least frequent chemical: DIMETHYLDIOCTYL
##  Least frequent chemical: DODECADIEN-1-OL
##  Least frequent chemical: ETHEPHON
##  Least frequent chemical: FLUENSULFONE
##  Least frequent chemical: GIBBERELLIC ACID
##  Least frequent chemical: HYDROGEN PEROXIDE
##  Least frequent chemical: PEROXYACETIC ACID
```

```
## Request failed [404]. Retrying in 1.2 seconds...
```

```
## Request failed [404]. Retrying in 5.4 seconds...
```

```
##  [1] "No GHS classification found for ALKYL. DIM. BENZ. AM"
##  [2] "No GHS classification found for CHLOROPICRIN"
##  [3] "No GHS classification found for CUPRAMMONIUM ACETATE"
##  [4] "No GHS classification found for DECYLDIMETHYLOCTYL"
##  [5] "No GHS classification found for DICHLOROPROPENE"
##  [6] "No GHS classification found for DIDECYL DIM. AMMON."
##  [7] "No GHS classification found for DIMETHYL DISULFIDE (DMDS)"
##  [8] "No GHS classification found for DIMETHYLDIOCTYL"
##  [9] "No GHS classification found for DODECADIEN-1-OL"
## [10] "No GHS classification found for ETHEPHON"
## [11] "No GHS classification found for FLUENSULFONE"
## [12] "No GHS classification found for GIBBERELLIC ACID"
## [13] "No GHS classification found for HYDROGEN PEROXIDE"
## [14] "No GHS classification found for PEROXYACETIC ACID"
```